BAB IV
HASIL DAN PEMBAHASAN

Pada bab ini dijelaskan mengenai implementasi metode yang telah dijelaskan pada bab metodologi penelitian, implementasi data, implementasi program yang digunakan, melakukan pengujian, dan penjelasan terkait hasil yang diperoleh pada saat implementasi.

4.1. Implementasi Data

Data yang digunakan dalam penelitian ini adalah data CSC Tangerang City dari Januari 2016 – September 2019 dengan format csv. Data yang digunakan hanya variabel terima. Jumlah data yang digunakan sebanyak 916. Dataset ini kemudian digunakan pada IDE Jupyter Notebook untuk melakukan prediksi pendapatan menggunakan metode LSTM. Berikut pada gambar 4.1 adalah potongan gambar dari data tersebut:

```
27680.0
46700.0
135200.0
86850.0
31950.0
681990.0
119500.0
413820.0
310810.0
59500.0
142820.0
229550.0
415110.0
307310.0
151080.0
203890.0
180090.0
366997.4
529280.0
```

Gambar 4.1 Potongan Dataset

4.1.1. *Difference* Data

Difference data menggunakan fungsi dari *numpy*.

```
diff_dataset = np.diff(dataset)
```

Gambar 4.2 *Source Code Difference* Data

4.1.2. *Scaling* Data

Scaling data menggunakan *MinMaxScaler*.

```
diff_dataset = np.reshape(diff_dataset, (-1, 1))
scaler = MinMaxScaler(feature_range=(0,1))
scaled = scaler.fit_transform(diff_dataset)
series = pd.DataFrame(scaled)
```

Gambar 4.3 *Source Code Scaling* Data

4.1.3. Uji Stasioneritas

Uji stasioneritas menggunakan *Dickey-Fuller Test*. Berikut pada gambar 4.4 adalah *source code* uji stasioneritas yang digunakan.

```
df2=df1.resample('D', how=np.mean)

def test_stationarity(timeseries):
    rolmean = timeseries.rolling(window=30).mean()
    rolstd = timeseries.rolling(window=30).std()

    plt.figure(figsize=(14,5))
    sns.despine(left=True)
    orig = plt.plot(timeseries, color='blue',label='Original')
    mean = plt.plot(rolmean, color='red', label='Rolling Mean')
    std = plt.plot(rolstd, color='black', label = 'Rolling Std')

    plt.legend(loc='best'); plt.title('Rolling Mean & Standard Deviation')
    plt.show()

    print ('<Results of Dickey-Fuller Test>')
    dftest = adfuller(timeseries, autolag='AIC')
    dfoutput = pd.Series(dftest[0:4],
                         index=['Test Statistic','p-value','#Lags Used','Number of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print(dfoutput)
test_stationarity(df2.Terima.dropna())
```

Gambar 4.4 *Source Code* Uji Stasioneritas

```
<Results of Dickey-Fuller Test>
Test Statistic                  -4.5422
p-value                          0.0002
#Lags Used                      11.0000
Number of Observations Used    904.0000
Critical Value (1%)             -3.4376
Critical Value (5%)             -2.8647
Critical Value (10%)            -2.5685
dtype: float64
```

Gambar 4.5 Hasil Uji Stasioneritas

Terlihat pada gambar 4.5 adalah hasil dari *Dicky-Fuller Test*. Untuk mengetahui data stasioner atau tidak dengan melihat Test Statistic dan p-value, apabila nilai Test Statistic lebih kecil dari Critical Value atau p-value lebih kecil sama dengan 0.05, maka data tidak memiliki unit root dan data stasioner. Data yang

digunakan memiliki nilai Test Statistic -4.5422 dan nilai p-value 0.0002, sehingga data yang digunakan stasioner.

4.2. Implementasi Program

Berikut adalah implementasi dari program dengan menggunakan IDE Jupyter Notebook dengan Bahasa *Python*:

4.2.1. *Moving Window*

Inisiasi dengan ukuran *moving window* 16. *Moving window* menggunakan fungsi *shift* dari *panda* untuk menggeser seluruh kolom dengan jumlah yang telah ditentukan. Pada gambar , menggeser kolom ke atas 1 dengan – (i+1). Jika ingin menggeser kolom ke bawah 1 menggunakan + (i+1) kemudian menggabungkan kolomnya ke data asli.

```
window_size = 16

series_s = series.copy()
for i in range(window_size):
    series = pd.concat([series, series_s.shift(-(i+1))], axis = 1)

series.dropna(axis=0, inplace=True)
np.isnan(series).sum().sum()
```

Gambar 4.6 *Source Code Moving Window*

4.2.2. Pembagian Data Train dan Data Test

Pada gambar 4.7 data dibagi dengan komposisi data train 90% dan data test 10%.

```
nrow = round(0.9*series.shape[0])

train = series.iloc[:nrow, :]
test = series.iloc[nrow:,:]

train_X = train.iloc[:,:-1]
train_y = train.iloc[:,-1]
test_X = test.iloc[:,:-1]
test_y = test.iloc[:,-1]

train_X = train_X.values
train_y = train_y.values
test_X = test_X.values
test_y = test_y.values

train_X = train_X.reshape(train_X.shape[0],train_X.shape[1],1)
test_X = test_X.reshape(test_X.shape[0],test_X.shape[1],1)
```

Gambar 4.7 *Source Code* Pembagian Data Train dan Data Test

4.2.3. Implementasi Model LSTM

Pembuatan model LSTM menggunakan *library* Keras. Model dari arsitektur LSTM dengan menggunakan 4 layer. Layer 1 dan 4 menggunakan 16 neuron. Layer 2 dan 3 adalah hidden layer menggunakan 128 neuron pada gambar 4.8. Hidden layer akan berubah sesuai dengan skema pengujian yang telah dibuat.

```
model = Sequential()
model.add(LSTM(input_shape = (window_size,1), output_dim= 16, return_sequences = True)) #layer 1
model.add(Dropout(0.5))
model.add(LSTM(128, return_sequences=True)) #layer 2
model.add(Dropout(0.5))
model.add(LSTM(128, return_sequences=True)) #layer 3
model.add(Dropout(0.5))
model.add(LSTM(16)) #layer 4
model.add(Dropout(0.5))
model.add(Dense(1))
model.add(Activation("linear"))
opt = Adam(lr=0.0001)
model.compile(loss="mse", optimizer=opt)
model.summary()
```

Gambar 4.8 *Source Code* Arsitektur LSTM

4.2.4. Implementasi Prediksi Pendapatan

Prediksi pada data train dan data test, kemudian mengembalikan nilai dari normalisasi. Hasil pengukuran prediksi menggunakan RMSE dan mencetak hasilnya.

```
preds_train = model.predict(train_X) #predict train data
preds_train = scaler.inverse_transform(preds_train)
actuals_train = scaler.inverse_transform(train_y.reshape(-1,1))
preds_test = model.predict(test_X) #predict test data
preds_test = scaler.inverse_transform(preds_test)
actuals_test = scaler.inverse_transform(test_y.reshape(-1,1))
print("RMSE Train Data: ", math.sqrt(mean_squared_error(actuals_train,preds_train)))
print("RMSE Test Data: ", math.sqrt(mean_squared_error(actuals_test,preds_test)))
```

Gambar 4.9 *Source Code* Prediksi

4.3. Pengujian

Pengujian dilakukan untuk mengukur suatu percobaan dan dalam penelitian ini yaitu akurasi hasil prediksi menggunakan RMSE. Pengujian dilakukan menggunakan 1 unit laptop dengan spesifikasi *Hardware*: Processor Intel Core i5-6200U CPU @ 2.30 GHz, Tipe Sistem 64-bit, Memory 4 GB, dan Sistem Operasi Windows 10 Pro. Pengujian dilakukan sesuai dengan rancangan pengujian. Setiap pengujian dilakukan dengan komposisi data train dan data test yang berbeda, kemudian mencari jumlah neuron hidden layer dan epoch yang paling optimal.

Setelah mendapatkan hyperparameter yang optimal, melakukan pengujian pada komposisi data train dan data test yang telah ditetapkan sekali lagi.

4.3.1. Pengujian komposisi data train 50% dan data test 50%

Pengujian jumlah neuron hidden layer dengan komposisi data train 50% dan data test 50%. Berikut adalah gambar dari sampel pengujian 512 neuron hidden layer:

```
Train on 405 samples, validate on 45 samples
Epoch 1/600
405/405 [==============================] - 9s 23ms/step - loss: 0.1866 - val_loss: 0.1040
Epoch 2/600
405/405 [==============================] - 6s 14ms/step - loss: 0.0706 - val_loss: 0.0292
Epoch 3/600
405/405 [==============================] - 6s 15ms/step - loss: 0.0583 - val_loss: 0.0292
Epoch 4/600
405/405 [==============================] - 6s 14ms/step - loss: 0.0441 - val_loss: 0.0394
Epoch 5/600
405/405 [==============================] - 6s 14ms/step - loss: 0.0524 - val_loss: 0.0366
Epoch 6/600
405/405 [==============================] - 6s 14ms/step - loss: 0.0465 - val_loss: 0.0289
Epoch 7/600
405/405 [==============================] - 6s 14ms/step - loss: 0.0440 - val_loss: 0.0289
```

Gambar 4.10 Pengujian 512 Neuron Data Train 50% dan Data Test 50%

Tabel 4.1 Kombinasi Jumlah Neuron Hidden Layer Data Train 50% dan Data Test 50%

No.	Neuron Hidden Layer	RMSE Train Data	RMSE Test Data
1.	16	829739.08	836596.13
2.	32	663686.33	662954.32
3.	64	652326.78	650307.98
4.	128	648466.19	648115.79
5.	256	671472.16	667542.49
6.	512	835291.95	841932.47

Berdasarkan tabel 4.1, menunjukan bahwa 128 neuron pada hidden layer memiliki nilai yang paling optimal pada komposisi data train 50% dan data test 50%.

4.3.2. Pengujian komposisi data train 60% dan data test 40%

Pengujian jumlah neuron hidden layer dengan komposisi data train 60% dan data test 40%. Berikut adalah gambar dari sampel pengujian 512 neuron hidden layer:

```
Train on 485 samples, validate on 54 samples
Epoch 1/600
485/485 [==============================] - 8s 17ms/step - loss: 0.1759 - val_loss: 0.0912
Epoch 2/600
485/485 [==============================] - 6s 12ms/step - loss: 0.0637 - val_loss: 0.0338
Epoch 3/600
485/485 [==============================] - 6s 12ms/step - loss: 0.0585 - val_loss: 0.0325
Epoch 4/600
485/485 [==============================] - 6s 12ms/step - loss: 0.0501 - val_loss: 0.0390
Epoch 5/600
485/485 [==============================] - 6s 12ms/step - loss: 0.0437 - val_loss: 0.0297
Epoch 6/600
485/485 [==============================] - 6s 12ms/step - loss: 0.0436 - val_loss: 0.0295
Epoch 7/600
485/485 [==============================] - 6s 12ms/step - loss: 0.0466 - val_loss: 0.0316
```

Gambar 4.11 Pengujian 512 Neuron Data Train 60% dan Data Test 40%

Tabel 4.2 Kombinasi Jumlah Neuron Hidden Layer Data Train 60% dan Data Test 40%

No.	Neuron Hidden Layer	RMSE Train Data	RMSE Test Data
1.	16	842007.35	802968.55
2.	32	689369.81	650965.41
3.	64	663039.68	624376.94
4.	128	662768.99	621805.69
5.	256	664147.06	623471.56
6.	512	853183.17	811925.42

Berdasarkan tabel 4.2, menunjukan bahwa 128 neuron pada hidden layer memiliki nilai yang paling optimal pada komposisi data train 60% dan data test 40%.

4.3.3. Pengujian komposisi data train 70% dan data test 30%

Pengujian jumlah neuron hidden layer dengan komposisi data train 70% dan data test 30%. Berikut adalah gambar dari sampel pengujian 512 neuron hidden layer:

```
Train on 566 samples, validate on 63 samples
Epoch 1/600
566/566 [==============================] - 9s 16ms/step - loss: 0.1686 - val_loss: 0.0751
Epoch 2/600
566/566 [==============================] - 6s 11ms/step - loss: 0.0595 - val_loss: 0.0298
Epoch 3/600
566/566 [==============================] - 7s 12ms/step - loss: 0.0482 - val_loss: 0.0346
Epoch 4/600
566/566 [==============================] - 7s 12ms/step - loss: 0.0524 - val_loss: 0.0304
Epoch 5/600
566/566 [==============================] - 7s 13ms/step - loss: 0.0476 - val_loss: 0.0252
Epoch 6/600
566/566 [==============================] - 7s 12ms/step - loss: 0.0461 - val_loss: 0.0291
Epoch 7/600
566/566 [==============================] - 7s 12ms/step - loss: 0.0445 - val_loss: 0.0266
```

Gambar 4.12 Pengujian 512 Neuron Data Train 70% dan Data Test 30%

Tabel 4.3 Kombinasi Jumlah Neuron Hidden Layer Data Train 70% dan Data Test 30%

No.	Neuron Hidden Layer	RMSE Train Data	RMSE Test Data
1.	16	726309.35	675763.78
2.	32	671473.14	615210.79
3.	64	664263.86	608313.83
4.	128	666300.75	613582.33
5.	256	686107.51	634269.87
6.	512	852227.04	795759.85

Berdasarkan tabel 4.3, menunjukan bahwa 64 neuron pada hidden layer memiliki nilai yang paling optimal pada komposisi data train 70% dan data test 30%.

4.3.4. Pengujian komposisi data train 80% dan data test 20%

Pengujian jumlah neuron hidden layer dengan komposisi data train 80% dan data test 20%. Berikut adalah gambar dari sampel pengujian 512 neuron hidden layer:

```
Train on 647 samples, validate on 72 samples
Epoch 1/600
647/647 [==============================] - 10s 15ms/step - loss: 0.1548 - val_loss: 0.0353
Epoch 2/600
647/647 [==============================] - 7s 12ms/step - loss: 0.0601 - val_loss: 0.0194
Epoch 3/600
647/647 [==============================] - 8s 12ms/step - loss: 0.0509 - val_loss: 0.0264
Epoch 4/600
647/647 [==============================] - 8s 12ms/step - loss: 0.0456 - val_loss: 0.0197
Epoch 5/600
647/647 [==============================] - 8s 12ms/step - loss: 0.0427 - val_loss: 0.0208
Epoch 6/600
647/647 [==============================] - 7s 12ms/step - loss: 0.0440 - val_loss: 0.0199
Epoch 7/600
647/647 [==============================] - 8s 12ms/step - loss: 0.0425 - val_loss: 0.0198
```

Gambar 4.13 Pengujian 512 Neuron Data Train 80% dan Data Test 20%

Tabel 4.4 Kombinasi Jumlah Neuron Hidden Layer Data Train 80% dan Data Test 20%

No.	Neuron Hidden Layer	RMSE Train Data	RMSE Test Data
1.	16	836121.90	802440.97
2.	32	677658.16	636603.26
3.	64	653162.59	613362.11
4.	128	653858.38	610696.59
5.	256	661585.48	615755.12
6.	512	840367.68	805776.17

Berdasarkan tabel 4.4, menunjukan bahwa 64 neuron pada hidden layer memiliki nilai yang paling optimal pada komposisi data train 80% dan data test 20%.

4.3.5. Pengujian komposisi data train 90% dan data test 10%

Pengujian jumlah neuron hidden layer dengan komposisi data train 90% dan data test 10%. Berikut adalah gambar dari sampel pengujian 512 neuron hidden layer:

```
Train on 728 samples, validate on 81 samples
Epoch 1/600
728/728 [==============================] - 10s 14ms/step - loss: 0.1425 - val_loss: 0.0314
Epoch 2/600
728/728 [==============================] - 8s 12ms/step - loss: 0.0516 - val_loss: 0.0276
Epoch 3/600
728/728 [==============================] - 8s 11ms/step - loss: 0.0494 - val_loss: 0.0312
Epoch 4/600
728/728 [==============================] - 9s 12ms/step - loss: 0.0477 - val_loss: 0.0270
Epoch 5/600
728/728 [==============================] - 8s 11ms/step - loss: 0.0443 - val_loss: 0.0272
Epoch 6/600
728/728 [==============================] - 8s 12ms/step - loss: 0.0410 - val_loss: 0.0261
Epoch 7/600
728/728 [==============================] - 9s 12ms/step - loss: 0.0415 - val_loss: 0.0282
```

Gambar 4.14 Pengujian 512 Neuron Data Train 90% dan Data Test 10%

Tabel 4.5 Kombinasi Jumlah Neuron Hidden Layer Data Train 90% dan Data Test 10%

No.	Neuron Hidden Layer	RMSE Train Data	RMSE Test Data
1.	16	749122.09	685113.87
2.	32	666843.22	599600.12
3.	64	650099.21	594401.29
4.	128	655327.92	595058.44
5.	256	651650.83	587911.37
6.	512	839514.28	759988.16

Berdasarkan tabel 4.5, menunjukan bahwa 64 neuron pada hidden layer memiliki nilai yang paling optimal pada komposisi data train 90% dan data test 10%. Pada prediksi data time series, tidak ada aturan pasti jumlah neuron hidden layer yang optimal karena tergantung dengan data yang digunakan pada percobaan tersebut. Sehingga untuk mendapatkan hasil yang optimal dapat dilakukan dengan percobaan.

4.3.6. Pengujian Parameter Epoch

Pengujian jumlah epoch menggunakan 128 neuron hidden layer dengan komposisi data train 50% dan data test 50% karena memiliki RMSE Train Data yang paling optimal dari komposisi yang lainnya.

```
Epoch 992/1000
405/405 [==============================] - 4s 9ms/step - loss: 0.0172 - val_loss: 0.0170
Epoch 993/1000
405/405 [==============================] - 4s 9ms/step - loss: 0.0155 - val_loss: 0.0176
Epoch 994/1000
405/405 [==============================] - 4s 9ms/step - loss: 0.0169 - val_loss: 0.0174
Epoch 995/1000
405/405 [==============================] - 4s 9ms/step - loss: 0.0149 - val_loss: 0.0171
Epoch 996/1000
405/405 [==============================] - 4s 9ms/step - loss: 0.0163 - val_loss: 0.0170
Epoch 997/1000
405/405 [==============================] - 4s 9ms/step - loss: 0.0158 - val_loss: 0.0172
Epoch 998/1000
405/405 [==============================] - 4s 9ms/step - loss: 0.0155 - val_loss: 0.0173
Epoch 999/1000
405/405 [==============================] - 4s 9ms/step - loss: 0.0162 - val_loss: 0.0174
Epoch 1000/1000
405/405 [==============================] - 4s 9ms/step - loss: 0.0160 - val_loss: 0.0171
```

Gambar 4.15 Pengujian Jumlah Epoch

Tabel 4.6 Kombinasi Jumlah Epoch

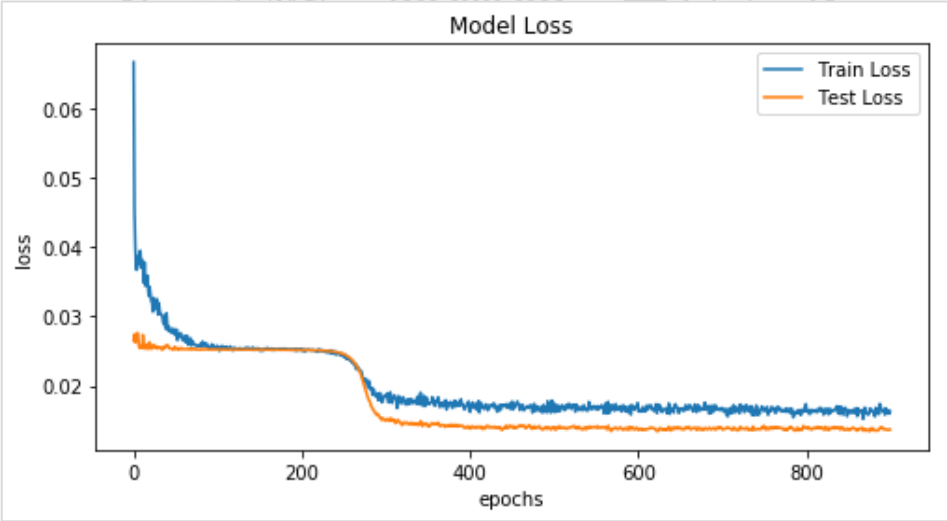
No.	Epoch	Train Loss	Test Loss
1.	100	0.027042	0.028801
2.	200	0.024540	0.028489
3.	300	0.024134	0.028478
4.	400	0.023547	0.028261
5.	500	0.021477	0.023836
6.	600	0.017006	0.017719
7.	700	0.016869	0.017520
8.	800	0.015518	0.017930
9.	900	0.015492	0.017585
10.	1000	0.015998	0.017115

Berdasarkan tabel 4.6, menunjukan bahwa jumlah 900 epoch memiliki hasil train loss yang paling optimal. Pada prediksi data time series, tidak ada aturan pasti jumlah epoch yang optimal karena tergantung dengan data yang digunakan pada percobaan tersebut. Sehingga untuk mendapatkan hasil yang optimal dapat dilakukan dengan percobaan.

Setelah mendapatkan jumlah neuron hidden layer dan epoch yang paling optimal, melakukan pengujian sekali lagi dengan kombinasi hyperparameter optimal tersebut.

Tabel 4.7 Kombinasi Hyperparameter Optimal

No.	Komposisi Data	Neuron Hidden Layer	Epoch	RMSE Train Data	RMSE Test Data
1.	data train 50% dan data test 50%	128	900	648408.28	649151.41
2.	data train 60% dan data test 40%	128	900	654693.06	618770.27
3.	data train 70% dan data test 30%	128	900	655382.25	603286.15
4.	data train 80% dan data test 20%	128	900	644197.87	606569.65
5.	data train 90% dan data test 10%	128	900	641375.70	594197.70

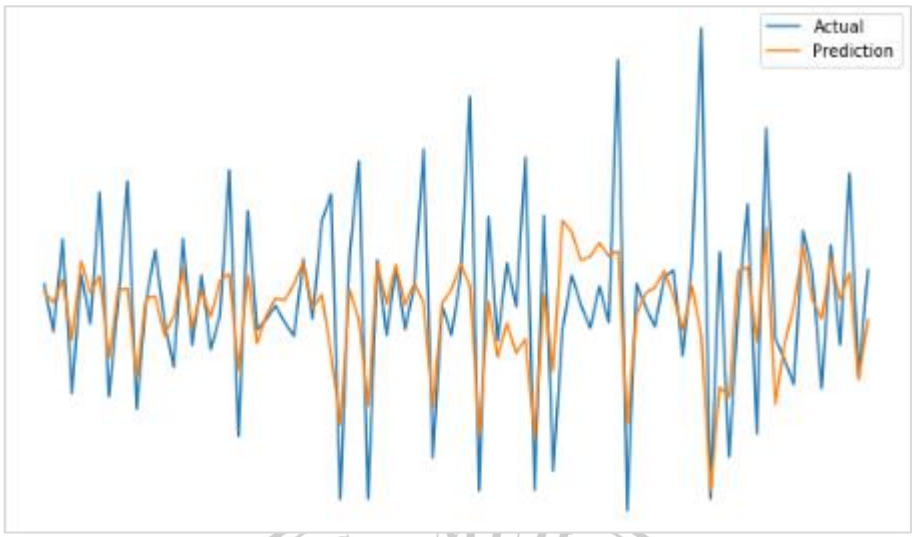


Gambar 4.16 Grafik Model Loss Data Train 90% dan Data Test 10%

Berdasarkan tabel 4.7, menunjukan bahwa hasil yang paling optimal adalah dengan hyperparameter 128 neuron hidden layer, 900 epoch, dan komposisi data train 90% dan data test 10%. Pada gambar 4.16 adalah grafik model loss dari kombinasi hyperparameter yang optimal tersebut.

Hasil prediksi perlu dilakukan perbandingan dengan data asli untuk mengukur kesalahan dari hasil prediksi. Pada gambar 4.17 adalah grafik perbandingan data asli dengan data hasil prediksi dan pada gambar 4.18 adalah

potongan nilai dari 5 data pertama hasil perbadingan data asli dengan data hasil prediksi.



Gambar 4.17 Grafik Data Asli dan Data Prediksi

	Actual	Prediction
0	152588.02	90688.54
1	-224666.00	4224.16
2	506798.88	185312.19
3	-709269.12	-290409.88
4	229877.89	331055.50

Gambar 4.18 Potongan Nilai Data Asli dan Data Prediksi

4.4. Analisa Hasil

Berdasarkan hasil pengujian prediksi menggunakan arsitektur LSTM mendapatkan hasil sebagai berikut:

a. Pada pengujian jumlah neuron hidden layer yang paling optimal yaitu pada pengujian komposisi data train 50% dan data test 50% dengan menggunakan 128 neuron hidden layer mendapatkan nilai RMSE data train sebesar 648466.19.
b. Pada pengujian jumlah epoch yang paling optimal yaitu pada 900 epoch dengan nilai train loss pada model sebesar 0.015492.
c. Pada pengujian dengan hyperparameter optimal mendapatkan hasil terbaik yaitu pada komposisi data train 90% dan data test 10% dengan nilai RMSE sebesar data train 641375.70 dan data test 594197.70. Terlihat pada gambar

4.16, grafik model loss yang dihasilkan cukup bagus karena nilai loss data train dan data test menurun seiring dengan epoch, serta loss data test yang tidak melebihi data train pada akhir yang menunjukkan bahwa model tidak *overfit*.

d. Nilai RMSE yang dihasilkan besar disebabkan karena beberapa alasan, yaitu pengujian RMSE dilakukan setelah model prediksi mengembalikan nilai dari scaling (range nilai 0-1) menjadi nilai normal sebelum scaling dan tergantung pada data yang digunakan. Data memiliki range nilai minimum -1633033 dan nilai maksimum 2166061.

e. Berdasarkan grafik perbandingan data prediksi dan data asli pada gambar 4.17, hasil prediksi memiliki akurasi cukup baik mendekati nilai aslinya meskipun sedikit dan masih banyak data prediksi yang jauh dari nilai aslinya. Tetapi, data prediksi memiliki nilai yang turun dan naik sama dengan nilai asli cukup banyak. Dengan tingkat akurasi kesalahan 16.67% pada nilai penurunan dan kenaikan data prediksi. Sebanyak 15 data dari 90 perbandingan data yang mengalami kesalahan yaitu, pada data ke-14 menuju ke-15, data ke-27 menuju data ke-28, data ke-44 menuju data ke-45, data ke-59 menuju data ke-60, data ke-65 menuju data ke-66, data ke-66 menuju data ke-67, data ke-80 menuju data ke-81, dan data ke-81 menuju data ke-82 yang seharusnya mengalami penurunan bukan kenaikan dan pada data ke-31 menuju data ke-32, data ke-34 menuju data ke-35, data ke-41 menuju data ke-42, data ke-46 menuju data ke-47, data ke-57 menuju data ke-58, data ke-68 menuju data ke-69, dan data ke-71 menuju data ke-72 yang seharusnya mengalami kenaikan bukan penurunan.